Differential evolution to solve the lot size problem in stochastic supply chain management systems

Kris Lieckens & Nico Vandaele

Annals of Operations Research

ISSN 0254-5330

Ann Oper Res
DOI 10.1007/s10479-014-1778-0

Springer

Ann Oper Res
DOI 10.1007/s10479-014-1778-0

Differential evolution to solve the lot size problem in stochastic supply chain management systems

Kris Lieckens · Nico Vandaele

Abstract An advanced resource planning model is presented to support optimal lot size decisions for overall performance improvement of real-life supply chain management systems in terms of either total delivery time or total setup costs. Based on a queueing network, a model is developed for a mix of products, which follow a sequence of operations taking place at multiple interdependent supply chain members. At the same time, various sources of uncertainty, both in demand and process characteristics, are taken into account. In addition, the model includes the impact of parallel servers for multiple resources with period dependent time schedules. The corrupting influence of variabilities from rework and breakdown is also explicitly modeled. This integer non-linear problem is solved by standard differential evolution algorithms. They are able to find each product's lot size that minimizes its total supply chain lead time. We show that this solution approach outperforms the steepest descent method, an approach commonly used in the search for optimal lot sizes. For problems of realistic size, we propose appropriate control parameters for an efficient differential evolutionary search process. Based on these results, we add a major conclusion on the debate concerning the convexity between lot size and lead time in a complex supply chain environment.

Keywords Differential evolution · Evolutionary optimization · Lot sizing · Production planning · Queueing network

1 Introduction

Even in today's state-of-the-art production systems that have implemented lean manufacturing techniques like 6-sigma and SMED analysis, the amount of time to switch the production

K. Lieckens (✉) · N. Vandaele
Faculty of Economics and Business, Research Center for Operations Management,
KU Leuven, Leuven, Belgium
e-mail: kris.lieckens@kuleuven.be

N. Vandaele
Faculty of Economics and Business, KU Leuven Campus Kortrijk, Kortrijk, Belgium
e-mail: nico.vandaele@kuleuven.be

Published online: 10 January 2015

process between different product types may remain significant. Furthermore, the focused factory principle, a strategy that is becoming even more important in case of difficult economic conditions, forces these production systems to limit their operations to the most critical, highest value-added activities. Many small, specialized companies at a local and global scale arise where the interdependent relationships require coordination. In such a network of closely cooperating supply chain members, internal operations at multiple site locations are connected by transportation links. Given this lean and focused approach, the overall performance of the supply chain with respect to total lead times and total setup costs can still be improved by optimizing the lot size quantities to be used for each product type that flow between the entry and exit point of this system (Karmarkar et al. 1985b). Since the impact of these lot size decisions on the lead time is a difficult operational problem, it has received much attention in production planning research (see below). It involves a trade-off between capacity and lead time.

Lot sizes affect lead time, work-in-process (WIP), safety stock, due date performance and costs because they affect resource utilization and processing times. These two factors have distinct, but opposite influences on the KPIs, leading to a convex relationship between the lot sizes and these KPIs (Karmarkar 1987). Since all these KPIs are interdependent by the lead time, we can illustrate this relationship for the lead time only. As the lot size quantity gets smaller, more setups consume more resource capacity. This creates a higher utilization level, because more time is required by the resource to deliver the same production volume. The utilization calculation [Eq. (4)] will also show that the total required time includes the setup time. Since utilization levels cannot exceed 100 %, there is a lower bound on the lot size. From queueing theory it is known that the lead time is a nonlinearly increasing function of utilization: it is rather stable at low levels, while it explodes at high levels of utilization (Hopp and Spearman 2000). This is referred to as the saturation effect and is visualized in Fig. 1 by the nonlinearly decreasing lead time as a function of the lot size. The other effect is the batching effect. As the lot size quantity gets smaller, the lead time benefits from two effects: lots being finished more rapidly (wait-in-batch-time) and lots being collected more rapidly before the initial operation starts (wait-to-batch-time). This is the linearly increasing function in Fig. 1. The net impact of the lot size on the lead time is described by adding these two opposing effects, leading to the U-shaped function in Fig. 1. It is clear that the saturation effect and batching effect are responsible for the convex relationship between lot size and lead time, and that an optimal lot size exists. A function (bold line) is convex if and only if the region above its graph (shaded area) is a convex set, which is a set of points containing all line segments between each pair of its points. We refer to Karmarkar (1987) for a mathematical derivation in case of a single server queue with

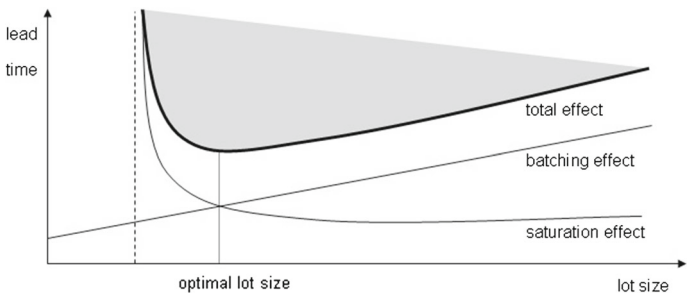

Fig. 1 Convex relationship between lot size and lead time

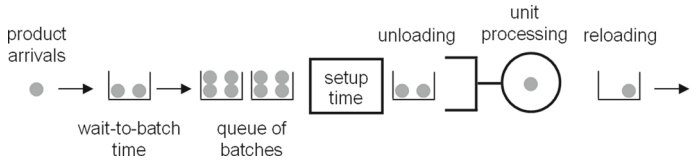

Fig. 2 Sequential production system

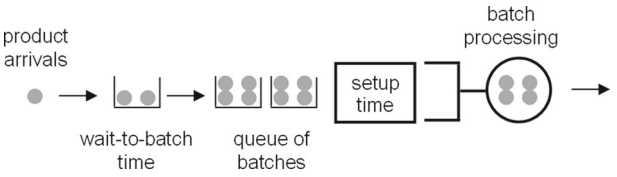

Fig. 3 Simultaneous production system

Markovian arrivals and Markovian processing times for a single product. This relationship typically applies to operations such as molding, painting, bottling, etc., where parts are sequentially processed (see Fig. 2). A delay due to setup is incurred before switching to a different product type and total process time grows proportionally with the lot size. This is in contrast to a simultaneous operation that works on several parts at a time like heating, and where a maximum number of parts can fit (see Fig. 3). Although here the process time is independent from the lot size, a similar convex relationship holds due to analogue saturation and batching effects. The analysis is only slightly different and therefore we focus on the sequential case. More details and applications can be found in Hopp and Spearman (2000).

It is clear that the lot size decision is a difficult problem, especially in supply chain management systems (SCMS) that are characterized by multiple products, multiple resources, multiple operations, multiple time schedules and various sources of process disruptions (e.g. stochastic demand, variabilities in production times, rework, breakdowns etc.). The goal of this paper is twofold. First of all, we want to develop a planning tool that supports optimal lot size decisions for product flows within a given SCMS. This type of decision is typically taken at the intermediate planning horizon. We further refer to this problem as Advanced Resource Planning (ARP). We choose for a queueing methodology to model this problem in an analytical way by using steady state relationships. Secondly, we want to show that standard versions of the differential evolution algorithm (DE) are able to find the global optimal values for lot sizes to be used in SCMS of realistic size. We contrast the performance of this evolutionary algorithm to the steepest descent algorithm (SD) because it has been selected in the literature for solving a similar ARP problem (Lambrecht et al. 1998). It will be clear that DE significantly outperforms the traditional approach of SD in terms of objective value and computation time. For an in-depth discussion of the state-of-the art in the area of aggregate planning models, in particular for those models that recognize the (nonlinear) relationship between the planned utilization of capacity and lead times, we refer to Pahl et al. (2007).

The model for ARP builds upon the work of Vandaele (1996) and Lambrecht et al. (1998), who use a similar approach for individual production systems. Given current intense cooperation between multiple specialized plants at a local and global scale, we want to extend this approach towards operations beyond the boundaries of a firm, while still including all its internal operations. We also relax the limitation of single resources by implementing

equations for parallel servers. Another adjustment relates to the integration of a quantitative measure for the impact of other operational issues like different time schedules, rework and breakdowns. In addition to the weighted average lead time as an overall performance measure, we develop an alternative objective where the lot size impact is translated into a financial measure that consists of costs for setup and inventory. We assume no uncertainty regarding these cost parameters. The approach of Zhou and Guan (2013) handles this issue in a single-item two-stage stochastic lot-sizing problem. The model for ARP is further described in Sect. 2.

The main focus of this paper is on the optimization procedure for the ARP model. It is difficult to solve for the lead time minimizing lot size for each product type because the nonlinear, interrelated equations with integer decision variables make it an NP-complete problem. This requires a heuristic search approach. In the literature, this problem is commonly handled by SD as described in Vandaele (1996) and applied in Lambrecht et al. (1998). Section 5 will show that the continuous treatment of discrete variables may lead to suboptimal, or even infeasible results. Since DE as a member of the broader family of genetic algorithms has proven to have a strong performance in solving hard and complex problems, even when many local optima exist and when the objective function is rather flat (Lampinen and Zelinka 1999; Storn and Price 1997; Babu and Angira 2002), we opt for this heuristic search method. For our problem settings, even standard DE schemes proof to improve the performance of SD significantly. Efficient control parameters that quickly lead to stable and reliable results are derived. These results are consistent for multiple independent runs with DE, which is an indication of a rather smooth objective function.

By further exploiting a specific search characteristic of DE, we are able to provide supportive evidence for the convex relationship between the lot size and the objective function, expressed in terms of either total lead time or total cost. This finding is the major contribution of the paper. Only for the single product case and for an M/M/1 queue, Karmarkar (1987) has described an analytical proof. For the multi-product, multi-resource case with multiple operations linked in the supply chain, the complexity of an analytical analysis of the lot size problem (mainly caused by interdependencies between total lead times and the mix of product types) increases in such a way that in this general case, there are no exact closed-form analytical expressions for the optimal lot size; only approximations exist. A mathematical proof of the convexity in this complex case does not exist either. In a multi-product, single stage setting with an M/G/1 queue, the expected queueing delay is quasi-convex in the lot size (Karmarkar et al. 1992). When lot size independent weights are used to obtain a weighted average total lead time of a batch, this lead time measure is also quasi-convex in the lot sizes (Kuik and Tielemans 2004). As the production system becomes more complicated (multiple stages, multiple resources, multiple servers, . . .), no proof on the convexity can be found in the literature. Authors who consider this kind of systems like Lambrecht et al. (1998) are limited to postulate the convex behavior.

The research questions, for which findings are outlined in Sect. 5, are summarized as follows:

1. Do standard schemes of the differential evolution algorithm exist to optimize large real-life ARP problems efficiently in a SCMS?
2. Is there evidence for the generally postulated convex relationship between lot size and lead time in a multi-product, multi-resource SCMS?

In Sect. 2 the characteristics and assumptions of the model are described in more detail. The steady state equations from queueing theory to estimate total lead time and total costs

are established in Sect. 3. The optimization routines are presented in Sect. 4, followed by the results for real-life applications in Sect. 5. Conclusions are outlined in Sect. 6.

2 Model description

The model for ARP is developed to support lot size decisions for distinct product flows that require processing by a sequence of multiple, interdependent members in a given SCMS. This mix of products is handled on multiple resources according to a first-come, first-serve priority rule. Resources can be either machines or labor, both with single or parallel servers. The demand is based on forecasts and firm customer orders, while the dynamic nature of the customer demand pattern is explicitly taken into account. Demand is independent and identically distributed, but we do not pose any restriction on the type of this distribution. Each product has its own specific bill of processes, with a fixed sequence of internal and external operations that are known prior to the batch release (i.e. deterministic routing). Since setup and production processes are typically stochastic, we assume that their required times are also generally distributed. The degree of randomness in arrival, setup and production processes are described in the model by a squared coefficient of variation (SCV). Another process characteristic is that a product's lot size quantity remains fixed along its route, between the entry and exit point of the SCMS. As a result, transfer batching is not allowed, which means that the entire batch must be completed before items are released to the next operation. Given the required coordination and information sharing between multiple, mutually dependent focused factories in the supply chain, it is reasonable to use the same, product specific lot size along its total routing. We further assume that setup times and setup costs are independent of the production sequence. This model can be executed in a repetitive and independent way for multiple planning periods at the intermediate time range, which makes it useful for the sales and operations planning process. These time buckets that constitute the planning horizon are long enough to ensure steady state conditions.

In order to determine the optimal lot sizes that minimize lead times or costs, we transform the SCMS into a queueing network with steady state equations where the parameters and the expected lead time are a function of the lot size (see Sect. 3). As a result, the utilization impact and the stochastic nature of the problem are included in the model, while lead times and inventory costs can be estimated in an analytical way (Karmarkar et al. 1985a).

The queueing approach is similar to Lambrecht et al. (1998), who on their turn have included some additional features into queueing approximations found in the literature (Shanthikumar and Buzacott 1981; Buzacott and Shanthikumar 1985; Shanthikumar and Sumita 1988; Bitran and Tirupati 1988). One of their main contribution is the approximation of the lead time distribution by a lognormal distribution, which is used to estimate a safety lead time such that customer orders are satisfied with a predefined service probability. Once the optimal lot sizes are determined, the corresponding minimal expected lead time can be obtained for each product. This is one of the main ARP outputs because it enables release date settings that will meet a specific due date under a given service level constraint by adding a safety lead time to the expected lead time. It creates a time window within which all the operations must be scheduled. Lambrecht et al. (1998) suggest a hierarchical approach to link separate applications into an integrated planning and scheduling system. In this paper, we only focus on the lot size optimization process (Sect. 4) and a refinement of the underlying queueing model (Sect. 3). More elaborations and applications of ARP can be found in Vandaele et al. (2002, 2003), Nieuwenhuyse and Vandaele (2006), Nieuwenhuyse et al. (2007, 2011).

3 Model formulation

In this section, all the required relationships are derived to formulate the objective function for lead time estimation. Since this function will depend on the lot size quantities, an optimization routine can be developed. The following indices are used: multiple products $p \in \{1, \ldots, P\}$, multiple resources $r \in \{1, \ldots, R\}$ and for each product p, multiple operations $o \in \{1, \ldots, O_p\}$. These operations may be performed by multiple members of the supply chain, including the transportation processes that connect them. We define a binary parameter ς_{por} that is equal to 1 if operation o for product p requires resource r, and 0 otherwise. Product parameters related to the complete lot size are complemented by a tilde (). Given different operational schemes (shift regime, working time, breakdowns, ...), there is a need for a common time unit (e.g. hours) for all the time dependent parameters in the queueing model.

3.1 Availability

Since ARP will be used in real-life applications, the issue of different time schedules in a given planning period, also referred to as the time bucket, must be handled. Clearly, the occurrence of customer demand does not coincide with the manufacturing operating hours, and takes neither overtime, days-off and absenteeism of personnel nor breakdowns and preventive maintenance of machines into account. Therefore, we express all queueing parameters on a common, continuous time scale (24 h per day, 7 days per week, etc.). The advantage of this approach is that the outcome of the queueing model is easy to interpret: the delay is expressed in a number of calendar days, regardless of the available time for the respective resources in the underlying production system. To this end, we calculate the availability measure A_r as the percentage of time that each resource r is available for processing. When the breakdown pattern of a resource r is described by a Mean-Time-To-Repair $MTTR_r$ and a Mean-Time-Between-Failures $MTBF_r$, while the schedule of each resource r is given for the planning period (total number of working time units WT_r, i.e. in normal and overtime regime, and total number of time units for preventive maintenance PM_r), we can propose the following equation for the resource availability during the time bucket with continuous length of CT time units

$$A_r = \frac{(WT_r - PM_r)\frac{MTBF_r E_r}{MTTR_r + MTBF_r}}{CT}$$

where E_r is an efficiency percentage to subtract various sources of productive time losses not covered by the other measures. The fraction $MTBF_r/(MTTR_r + MTBF_r)$ represents the probability that a resource r is not failing. This fraction is applied to the net available working time of $WT_r - PM_r$. When compared to CT, we obtain the probability that resource r is available for value-added operations. The A_r-value may differ for different time buckets in the planning horizon.

3.2 Arrival process

When λ_p is the product specific demand quantity that we expect to arrive per time unit defined on a continuous time scale, then the expected time between demand occurrences is $IA_p = 1/\lambda_p$. The variability of the interarrival time for each product p is described by a SCV equal to $c^2_{IA_p}$. It only affects resource r of the first operation ($o = 1$). Subsequent operations

will face a variability at their inbound that is determined by variability and utilization levels at all upstream stations. Since this requires modeling information from the production process, it is derived in Sect. 3.3.

Before these demand arrivals of product p are released into the shop floor, they are grouped by the manufacturing lot size Q_p, which we assume to remain constant over all operations o. This batching process has several effects. First of all, each product p is characterized by an average batch arrival rate $\tilde{\lambda}_p = \lambda_p/Q_p$ and a SCV of batch interarrival times $\tilde{c}^2_{IA_p} = c^2_{IA_p}/Q_p$. Another issue is the so-called wait-to-batch-time $WTBT_p$, a collection time to form a batch of size Q_p. This delay is only observed before the first operation as a batch is assumed not to be split when proceeding along the route. It is equal to

$$WTBT_p = \frac{(Q_p - 1)}{2\lambda_p}$$

because the first unit in a batch waits for $Q_p - 1$ other units to arrive and therefore waits $(Q_p - 1)/\lambda_p$ time units, whereas the last one does not have to wait at all to join the batch. A similar approach can be found in Vandaele et al. (2003).

For our queueing network approach, these multi-product batch arrival processes have to be aggregated into a single batch arrival process at each resource r. This aggregate process is also characterized by an average aggregate batch arrival rate $\tilde{\lambda}_r$ and a SCV of the aggregate batch interarrival times $\tilde{c}^2_{IA_r}$. When we define the aggregate batch arrival rate of product p at resource r as $\tilde{\lambda}_{pr} = \sum_o \tilde{\lambda}_p \varsigma_{por}$, we have $\tilde{\lambda}_r = \sum_p \tilde{\lambda}_{pr}$. This includes batch arrivals at resource r that are both internal to the SCMS, i.e. coming from a resource that can be located at any supply chain member, and external to the SCMS, i.e. coming from the customer. The external aggregate batch arrival rate at resource r is defined as $\tilde{\lambda}'_r = \sum_p \tilde{\lambda}_p \varsigma_{p1r}$. The value of $\tilde{c}^2_{IA_r}$ is derived in (5) in Sect. 3.3 because it depends on the variability of upstream production processes. At this moment, we can only obtain an approximation for the SCV of the external aggregate batch interarrival times

$$\tilde{c}'^2_{IA_r} \begin{cases} \approx \frac{1}{3} + \frac{2}{3} \sum\limits_p \frac{\tilde{\lambda}_p \varsigma_{p1r}}{\tilde{\lambda}'_r} \tilde{c}^2_{IA_p} & \text{if } \sum\limits_p \varsigma_{p1r} \geq 2 \\ = \tilde{c}^2_{IA_p} & \text{if } \sum\limits_p \varsigma_{p1r} = 1 \end{cases} \tag{1}$$

where the weights $1/3$ and $2/3$ are discussed in Lambrecht et al. (1998). It is a specific case of a general approximation found by Albin (1981).

3.3 Production process

Before being shipped to the customer, each product p typically requires some operations o that are performed on resources r. Each resource r can have multiple, parallel servers s_r. We allow for resource recurrences along a product's route. The resource r to be used for a particular operation o of product p is indicated by $\varsigma_{por} = 1$. Next, we list for each product p and operation o the following production characteristics: expected setup time SU_{po} with SCV $c^2_{SU_{po}}$ and variance $\sigma^2_{SU_{po}}$, expected unit processing time PR_{po} with SCV $c^2_{PR_{po}}$ and variance $\sigma^2_{PR_{po}}$, expected unit process rate $\mu_{po} = 1/PR_{po}$, and a rework percentage $rwrk_{po}$. The average and variance of these setup and process times need to be transformed into effective measures using the parameters A_r and $rwrk_{po}$. They are obtained as follows Hopp and Spearman (2000)

$$SUe_{po} = SU_{po} \Big/ \left(\sum_r A_r \varsigma_{por} \right)$$

$$PRe_{po} = PR_{po} \Big/ \left((1 - rwrk_{po}) \sum_r A_r \varsigma_{por} \right)$$

$$\sigma^2_{SUe_{po}} = c^2_{SU_{po}} SUe^2_{po}$$

$$\sigma^2_{PRe_{po}} = \frac{2PRe_{po} \sum_r MTTR_r \, (1 - A_r) \, \varsigma_{por} c^2_{PR_{po}} PRe^2_{po}}{1 - rwrk_{po}} + \frac{rwrk_{po} PRe^2_{po}}{(1 - rwrk_{po})^2}$$

Similar to the arrival process, we need an average and a SCV of the aggregate batch processing time on resource r. The expression for the average is a weighted average over all the products and all their operations on resource r

$$\begin{aligned} \tilde{PR}_r &= 1/\tilde{\mu}_r \\ &= \sum_p \frac{\tilde{\lambda}_{pr}}{\tilde{\lambda}_r} \sum_o \frac{\tilde{\lambda}_{p\varsigma por}}{\tilde{\lambda}_{pr}} TPRe_{po} \end{aligned} \tag{2}$$

where $\tilde{\lambda}_{pr}/\tilde{\lambda}_r$ is the probability that a randomly selected batch at the inbound of resource r belongs to product p, while $TPRe_{po}$ is the total production time at operation o of a batch with products p, including setup time, i.e. $TPRe_{po} = SUe_{po} + Q_p PRe_{po}$. The expression for the SCV is Lambrecht et al. (1998)

$$\tilde{c}^2_r = \sum_p \sum_o \frac{\tilde{\lambda}_{p\varsigma por}}{\tilde{\lambda}_r} TPRe^2_{po} \tilde{\mu}^2_r - 1 + \sum_p \sum_o \frac{\tilde{\lambda}_{p\varsigma por}}{\tilde{\lambda}_r} \frac{\sigma^2_{SUe_{po}} + Q_p \sigma^2_{PRe_{po}}}{TPRe^2_{po}} \tag{3}$$

Next, we determine the resource utilization ρ_r, or the adapted traffic intensity, which must be lower than 100 %

$$\begin{aligned} \rho_r &= \frac{\tilde{\lambda}_r}{\tilde{\mu}_r s_r} \\ &= \sum_p \sum_o \tilde{\lambda}_{p\varsigma por} TPRe_{po}/s_r \leq 1 \end{aligned} \tag{4}$$

The final input parameter to be derived for the queueing network is $\tilde{c}^2_{IA_r}$, the SCV of the aggregate batch interarrival times at resource r, for which the variability of the interdeparture times of batches leaving the upstream resource r' is to be known. Although several approximations for systems with multiple servers are presented in Buzacott and Shanthikumar (1993), it is reasonable to estimate it by the following linking equation (Hopp and Spearman 2000)

$$\tilde{c}^2_{ID_{r'}} \approx 1 + (1 - \rho^2_{r'}) \left(\tilde{c}^2_{IA_{r'}} - 1 \right) + 1 + \frac{\rho^2_{r'}}{\sqrt{s_{r'}}} \left(\tilde{c}^2_{r'} - 1 \right) \tag{5}$$

It depends not only on the utilization and variabilities at resource r', but also on the lot size decisions due to (1), (3) and (4). The arrival process at a downstream resource r is also influenced by the fraction of the product flow leaving resource r' and going towards resource r, i.e. $f_{r'r}$ as calculated below in (8). The SCV of the aggregate batch interarrival time at resource r for products coming from resource r' is related to $\tilde{c}^2_{ID_{r'}}$ according to Shanthikumar and Buzacott (1981)

$$\tilde{c}^2_{IA_{r'r}} = f_{r'r} \tilde{c}^2_{ID_{r'}} + (1 - f_{r'r}) \tag{6}$$

This is part of the weighted average expression for $\tilde{c}^2_{IA_r}$

$$\tilde{c}^2_{IA_r} = \sum_{r'} \left(\frac{\tilde{\lambda}_{r'}}{\tilde{\lambda}_r} f_{r'r} \right) \tilde{c}^2_{IA_{r'r}} + \frac{\tilde{\lambda}'_r}{\tilde{\lambda}_r} \tilde{c}'^2_{IA_r}$$

which by using (6) becomes

$$= \sum_{r'} \left(\frac{\tilde{\lambda}_{r'}}{\tilde{\lambda}_r} f_{r'r} \right) \left(f_{r'r} \tilde{c}^2_{ID_{r'}} + (1 - f_{r'r}) \right) + \frac{\tilde{\lambda}'_r}{\tilde{\lambda}_r} \tilde{c}'^2_{IA_r}$$

and when combined with (5) results in

$$\sum_{r'} \left(\frac{\tilde{\lambda}_{r'}}{\tilde{\lambda}_r} f_{r'r} \right) \left(f_{r'r} \left[1 + \left(1 - \rho^2_{r'}\right) \left(\tilde{c}^2_{IA_{r'}} - 1 \right) + \frac{\rho^2_{r'}}{\sqrt{s_{r'}}} \left(\tilde{c}^2_{r'} - 1 \right) \right] + (1 - f_{r'r}) \right) + \frac{\tilde{\lambda}'_r}{\tilde{\lambda}_r} \tilde{c}'^2_{IA_r}$$

Reorganization of this equation, and expressed as an equality, leads to R linear equations with R unknown parameters $\tilde{c}^2_{IA_r}$

$$\begin{aligned} &- \sum_{r'} \tilde{\lambda}_{r'} f^2_{r'r} \left(1 - \rho^2_{r'}\right) \tilde{c}^2_{IA_{r'}} + \tilde{\lambda}_r \tilde{c}^2_{IA_r} \\ &= \sum_{r'} \tilde{\lambda}_{r'} f_{r'r} \left(f_{r'r} \rho^2_{r'} \left(1 + \frac{\tilde{c}^2_{r'} - 1}{\sqrt{s_{r'}}} \right) + 1 - f_{r'r} \right) + \tilde{\lambda}'_r \tilde{c}'^2_{IA_r} \end{aligned} \tag{7}$$

To solve these equations, we have to obtain the transition matrix F. By using the assumed deterministic routings, the transitions consist of

1. the proportion of batches from outside the system (stage 0) and directed to the first resource in the routing (stage r)

$$f_{0r} = \tilde{\lambda}'_r / \sum_r \tilde{\lambda}'_r$$

2. the proportion of batches from resource r' (stage r') and directed to the next resource in the routing (stage r)

$$f_{r'r} = \sum_p \sum_o^{O_p - 1} \tilde{\lambda}_p \varsigma_{por'} \varsigma_{p(o+1)r} / \tilde{\lambda}_{r'} \tag{8}$$

3. the proportion of batches from the last resource in the routing (stage r) and directed to the outside of the system (stage 0)

$$f_{r'0} = \sum_p \tilde{\lambda}_p \varsigma_p o_{pr'} / \tilde{\lambda}_{r'}$$

At this point we are able to estimate the expected lead time of product p at an operation o at resource r. Instead of using the Kraemer–Lagenbach–Belz approximation as in Lambrecht et al. (1998), we opt for the approximation from Whitt (1993). The reason is twofold: it is a GI/G/m-model that applies to a more realistic production system with parallel servers s_r and it estimates the waiting time under heavy traffic conditions more accurately due to a

correction factor ϕ. By using (2), (3), (4) and (7), we can formulate the unit waiting time of product p in the queue at operation o as:

$$EW_{po} \approx \sum_r EWQ_r \varsigma_{por} + TPRe_{po}$$

with

$$EWQ_r \approx \phi\left(\rho_r, \tilde{c}^2_{IA_r}, \tilde{c}^2_r, s_r\right)\left(\frac{\tilde{c}^2_{IA_r} + \tilde{c}^2_r}{2}\right)\left(\frac{\rho_r^{\sqrt{2(s_r+1)}-1}}{s_r(1-\rho_r)}\right)\tilde{PR}_r$$

Since multiple products p visit resource r through one or more operations o, an expected aggregate lead time at resource r can be determined as a weighted average

$$EW_r \approx EWQ_r + \sum_p \sum_o \frac{\lambda_p \varsigma_{por}}{\sum_p \sum_o \lambda_p \varsigma_{por}} TPRe_{po} \tag{9}$$

3.4 Objective functions

We present two objective functions, an operational one and a financial one, subjected to a minimization routine. The first objective is an expected aggregate lead time for the overall production system

$$EW \approx \sum_r EW_r + \sum_p \frac{\lambda_p}{\sum_p \lambda_p} WTBT_p \tag{10}$$

The goal is to find a set of lot size values Q_p for each product p that leads to an optimal performance of the entire SCMS from a lead time perspective. However, when costs are involved for holding inventory and setting up resources, the optimal lot size decisions may be different because of a shifted trade-off function. We therefore propose a second objective that incorporates the lead time information from (1) and (9) into an overall cost function that consists of total setup costs and total lead time related costs. To this end, we list some additional cost parameters: the average cost hc_{po} to hold one unit in stock of product type p at the inbound of operation o during the predefined time bucket of length CT, a fixed setup cost suc_{po} each time the resource related to an operation o of product p is changed over, a number of operators l_{po} required for this setup process and a labor cost lbc for every lbt time units that are consumed for setting up resources. The second, alternative objective function in terms of the expected costs becomes

$$EC \approx \sum_p \lambda_p hc_{p1} WTBT_p + \sum_p \sum_o \lambda_p hc_{po} \sum_r EW_r \varsigma_{por} + CT \sum_p \sum_o \tilde{\lambda_p} suc_{po} + lbc \left\lfloor \frac{CT}{lbt} \sum_p \sum_o \tilde{\lambda_p} SU_{po} l_{po} + 1 \right\rfloor \tag{11}$$

It consists of two cost types: expected inventory costs and expected setup costs. The inventory costs are obtained by Little's Law: we apply expected arrival rates of product p to expected delays like expected WTBT at the first operation and expected queue times at resources r for each operation o. Little's Law is a basic and fundamental long-term relationship between inventory (WIP), throughput rate (arrival rate) and flow time (lead time) of any process in steady state: Inventory = Throughput × Flow Time (Little 1961). The fixed

setup cost suc_{po} at operation o is multiplied by the total number of setups in the planning period, i.e. CT time units multiplied by the number of batches of product p per time unit. We also take into account total labor time and total labor cost associated with performing setups. Since not all operators may be involved during the entire setup process that takes SU_{po} time units, l_{po} is allowed to be a non-integer value. It is assumed that a fixed labor cost lbc is charged for each integer multiple of lbt time units required by all the operators for setting up resources. This is the case when for instance workers are paid full-time. All other cost elements like depreciation, raw materials, disposal, repair, etc. are considered to be irrelevant to the lot size decision.

4 Model optimization

The goal of this paper is to determine for each product p an optimal value for its lot size Q_p that minimizes either the overall expected aggregate lead time as defined in (10) or the expected total costs as defined in (11). The model for ARP as described in Sect. 3 can be classified as an integer nonlinear programming problem (INLP). Some characteristics make it difficult to solve:

- Decision variables Q_p are discrete and must be selected within a user defined lower and upper bound (Q_p^{min} and Q_p^{max}), which constitute physical constraints (for instance limited number of available transportation carriers).
- Expected waiting times in the objective functions (10) and (11) are non-linearly dependent on the utilization level ρ_r.
- Since ρ_r depends on $\tilde{\lambda}_p$, which is equal to λ_p / Q_p, the R constraints in (4) are non-linearly dependent on Q_p.
- There are multiple conditional relationships in $\phi(\rho_r, \tilde{c}^2_{IA_r}, \tilde{c}^2_r, s_r)$.
- The introduction of a staircase function in (11) adds complexity to the search space because of the danger of getting trapped in a local optimum.

The computational complexity not only grows exponentially with the number of discrete variables and the number of decisions within each discrete variable, but also with the number of nonlinear relationships in the model. We compare two different algorithms to solve these INLP problems, a traditional method versus a genetic method that will lead to better results:

- The steepest descent method (SD). This is a deterministic search procedure because for a given set of algorithm control parameters, the next step of computation is always exactly known and determined.
- The differential evolution method (DE). This is a stochastic search procedure because for a given set of algorithm control parameters, the next step of computation is always unknown and undetermined.

We refer to Vandaele (1996) for more details on SD, while some standard DE routines are outlined next. Their optimization performance is investigated in Sect. 5.

4.1 Differential evolution routine

DE is an improved version of a genetic algorithm. It also manipulates populations based on the principle of survival of the fittest, but in contrast to genetic evolutionary algorithms, which use a predefined distribution function to drive the mutation, DE uses the difference of randomly sampled pairs of object vectors in such a way that the mutation reflects information

of the objective function it is optimizing. Instead of using only local information for each object vector, DE mutates all object vectors with the same universal distribution. The idea is to cover the entire search space in order to find a global optimum.

The method is defined as a parallel direct search method which operates on a population P_G of constant size that is associated with each generation G and consists of N vectors or candidate solutions $\mathbf{Q}_{n,G}$. Each vector $\mathbf{Q}_{n,G}$ contains all the decision variables, i.e. the P lot sizes Q_p in our problem. An individual lot size value for product p in the population is indicated by $Q_{p,n,G}$. This is briefly summarized as

$$P_G = \{\mathbf{Q}_{1,G}, \mathbf{Q}_{2,G}, \ldots, \mathbf{Q}_{n,G}, \ldots, \mathbf{Q}_{N,G}\}$$
$$\mathbf{Q}_{n,G} = [Q_{1,n,G}, Q_{2,n,G}, \ldots, Q_{p,n,G}, \ldots, Q_{P,n,G}]$$

$\forall n \in \{1, \ldots, N\}$, $\forall p \in \{1, \ldots, P\}$, $\forall G \in \{1, \ldots, G_{max}\}$ and $N \geq 4$. The DE steps are: selection of the strategy, initialization of the control parameters and the population, mutation and recombination to create new children, checking upper and lower bounds and building a new generation. The last step consists of constraint handling and evaluation of trial vectors.

4.1.1 Strategy

Although different strategies of DE exist (Storn and Price 1997; Zhang and Sanderson 2009), we will choose from the DE/rand/1/bin, DE/rand/2/bin and the DE/current-to-rand/1 schemes, which are explained in Sect. 4.1.4. The goal is to demonstrate that even these standard DE schemes can further improve the ARP objective functions in an effective and efficient way when compared to the results from SD as the traditional method for the ARP problem.

4.1.2 Control parameters

The user-defined control parameters, which remain constant during the search process, are the crossover constant CR, the population size NP, the mutation scaling factor F, the coefficient of combination K and the maximum number of generations G_{max}. Their meaning and appropriate values are explained below.

4.1.3 Population

To create the initial population $P_{G=0}$, a different value for each decision variable is randomly generated within its bounds according to

$$Q_{p,n,G=0} = Q_p^{min} + \text{rand}_p\,[0, 1]\left(Q_p^{max} - Q_p^{min} + 1\right) \tag{12}$$

where $\text{rand}_p\,[0, 1]$ represents a uniformly distributed random variable that ranges from zero to one. Since the outcome of (12) is a continuous value for the discrete variables Q_p, it is followed by truncation (i.e. rounding down) before the objective function is evaluated. Nevertheless, the underlying continuous values are still being used to create subsequent trial vectors. This procedure maintains the diversity of the population and the robustness of the algorithm because only feasible solutions give feedback to the optimization process (Lampinen and Zelinka 1999). This is in contrast to the SD procedure, which treats all the discrete variables as continuous during the optimization process and only rounds-off when the search process is finished. This may result in both infeasible and suboptimal solutions, as well as large deviations in the optimal objective function value. Generating infeasible vectors

is not problematic at this initial stage because it is the mutation based on the difference of vectors in combination with the constraint handling method in Sect. 4.1.6 that drives the population towards better solutions. Infeasible solutions are able to improve other, good and feasible solutions.

4.1.4 Mutation and recombination

Mutation aims to keep a population robust and to search a new area. DE mutates an object vector by adding the weighted difference of randomly sampled pairs of vectors in the current population P_G. The mutated vector that will be used to build the population for the next generation is denoted by $\mathbf{V}_{n,G+1}$.

Recombination, or crossover, is complementary to mutation and builds trial vectors out of existing object vector parameters in order to reinforce prior successes. The crossover operation creates a trial vector $\mathbf{U}_{n,G+1}$ by selecting elements from the target vector $\mathbf{Q}_{n,G}$ and the mutated donor vector $\mathbf{V}_{n,G+1}$. The population of child or trial vectors $P'_{G+1} = \mathbf{U}_{n,G+1} = U_{p,n,G+1}$ for each parent or target vector $\mathbf{Q}_{p,n,G}$ is created as follows

$$U_{p,n,G+1} = \begin{cases} V_{p,n,G+1} & \text{if } R(0) \leq CR \vee p = k \\ Q_{p,n,G} & \text{otherwise} \end{cases} \quad \forall n, \forall p \tag{13}$$

with $k \in \{1, \ldots, P\}$ randomly selected, $R(0) \in [0, 1]$ a uniformly random number and $CR \in [0, 1]$ the predefined crossover constant. CR controls the probability that a trial vector parameter will come from the mutated vector $\mathbf{V}_{n,G+1}$, instead of from the current vector $\mathbf{Q}_{n,G}$. We will choose from three basic mutation schemes: DE/rand/1/bin, DE/rand/2/bin and DE/current-to-rand/1.

In the *DE/rand/1/bin* scheme (further referred to as Scheme 1), we have $V_{p,n,G+1} = Q_{p,r_3,G} + F\left(Q_{p,r_1,G} - Q_{p,r_2,G}\right)$ in (13) with r_1, r_2, r_3 randomly selected from the N vectors and $r_1 \neq r_2 \neq r_3 \neq n$, $F \in (0, 1+]$ and $N > 3$. The $/1/$ in this scheme means that there is one paired difference of randomly chosen vectors that drives the mutation. Effective values for the weight F that scales the step size belongs to the interval $(0, 1+]$, i.e. values slightly larger than one can be used if needed, but do not appear to be productive (Lampinen and Zelinka 1999). The notation /rand/ means that the donors to be mutated are randomly chosen from the population members. Also, the randomly chosen indices r_1, r_2 and r_3 must be mutually different, and different from the current parent object vector n. Consequently, N must be larger than 3. New, random, integer values for r_1, r_2 and r_3 are chosen for each individual candidate solution n. The index k ensures that each child vector will differ from its parent in the previous generation by at least one variable. A new random integer value is assigned to k prior to the construction of each child vector. The binomial scheme $/bin/$ takes parameters from $\mathbf{V}_{n,G+1}$ each time when $R(0) \leq CR$, otherwise the parameters come from $\mathbf{Q}_{n,G}$.

The population of children in the *DE/rand/2/bin* scheme (further referred to as Scheme 2) uses two paired differences of randomly chosen vectors that drive the mutation. It is created with $V_{p,n,G+1} = Q_{p,r_5,G} + F(Q_{p,r_1,G} + Q_{p,r_2,G} - Q_{p,r_3,G} - Q_{p,r_4,G})$ in (13) while $r_1, r_2, r_3, r_4, r_5 \in \{1, 2, \ldots, N\}$ are randomly selected but $r_1 \neq r_2 \neq r_3 \neq r_4 \neq r_5 \neq n$ and $N > 5$.

In the *DE/current-to-rand/1* (further referred to as Scheme 3), child vectors are also created according to (13) with $V_{p,n,G+1} = Q_{p,n,G} + K\left(Q_{p,r_3,G} - Q_{p,n,G}\right) + F\left(Q_{p,r_1,G} - Q_{p,r_2,G}\right)$ and $K \in [-0.5, 1.5]$. The coefficient of combination K is a user-specified parameter that also controls the step size like F. CR does not need to be specified

because it is implicitly equal to 1. When $K = 1$, it is equivalent to Scheme 1 with $CR = 1$. When $K = 0$, there is only a simple mutation.

Candidate values for the control parameters are given by the guidelines of Storn and Price (1997):

- N should be as small as possible to limit the computation time. $N = 20P$ is appropriate for problems with $P \leq 30$. It can go up to $100P$ for functions with many local optima. When P is large or in case of few local optima, very good results can also be obtained with $N \ll 20P$. However, with $N < 2P$, all the object vectors in the population may tend towards a single, sub-optimal solution (i.e. premature convergence), or no evolution takes place in a still diversified population (i.e. stagnation).
- CR should be close to or equal to 1.
- F and K are closely related. Although K is continuous in the interval $[-0.5, 1.5]$, the range can be limited to $[0, 1]$ for practical problems. Mostly, it is even sufficient to set it to one of the three discrete values: 0, 0.5, 1. When $K \geq 0.5$, F should be in the range $[0.6, 0.8]$; smaller values of F may induce premature convergence. $F > 1$ seems not effective.

In case of premature convergence or stagnation, N and/or F should be increased. When Scheme 3 is used, K should be decreased to avoid premature convergence, or randomly chosen from the interval [0,1] to avoid stagnation.

4.1.5 Boundary check

A parameter of a mutated child vector that fails to lie within the boundary limits is randomly re-set between its original parent value and the violated boundary

$$U_{p,n,G+1} = \begin{cases} Q_p^{min} + R\,(0)\left(Q_{p,n,G} - Q_p^{min}\right) & \text{if } U_{p,n,G+1} < Q_p^{min} \\ Q_{p,n,G} + R\,(0)\left(Q_p^{max} - Q_{p,n,G}\right) & \text{if } U_{p,n,G+1} > Q_p^{max} \\ U_{p,n,G+1} & \text{otherwise} \end{cases}$$

4.1.6 Next generation

To select the vectors for the next generation, a one-to-one competition between each child and its parent is required, first on the level of the model constraints and then on the level of their performance according to the objective function.

Constraints limit the feasible solutions to a subset of the total search space. The R constraints from (4) in ARP are reformulated as being greater than zero when violated: $\rho_r - 1 \leq 0$. Instead of implementing them as 'soft' constraints by means of penalty functions, which may result in suboptimal or even infeasible solutions due to inappropriate values for the penalty parameters, we prefer to use the alternative constraint handling method from Lampinen (2001) to select the vectors for the next generation $\mathbf{Q}_{n,G+1}$. The advantage of this method is that the objective function is not evaluated until all the constraints are feasible, which reduces computation time and results in a fast convergence towards the feasible regions of the search space. In addition, the routine will not always have to go through all the constraints, which further decrease computation time. This is interesting for our computationally expensive objective and constraint functions.

Comparing a child to only one individual, its parent, and not to an arbitrary better performing member like in other evolutionary algorithms, ensures that there is no drift towards

local optima. Trial vectors that are equally good enter the population to avoid stagnation. The final step is to determine the best population member.

5 Model results

The steepest descent (SD) method, traditionally applied to ARP, is used as the base algorithm. The control parameters are a gradient precision to decide when an element of the gradient vector can be considered small enough to be zero, a step size precision to define the multiplicator applied to the gradient vector, an initial value for this step size, and a stopping precision to decide when an objective function improvement can be considered small enough to quit the search process. Apart from finding appropriate values for these parameters, the main problem with SD is that discrete variables are treated as continuous values. Rounding after optimization may not only lead to sub-optimal results, but also to even infeasible solutions. This is particularly true in ARP, where rounding down a lot size Q_p involves the risk of creating an overutilized resource r, especially in case of relatively small lot size values, high utilization levels and/or high setup times. Therefore, we consistently round up the lot size values found by SD. In addition, SD assumes continuity between every two points in the solution space, whereas the objective has conditional relations and staircase functions. SD implicitly handles this complication by a numerical first derivative, which is generated by exploring the feedback of the objective function in a local area around the current solution member. It determines the direction of improvement, which continues until no significant improvements can be made further.

The new algorithm applied to ARP is the differential evolution algorithm (DE). Apart from choosing a scheme (1=*DE/rand/1/bin*, 2=*DE/rand/2/bin*, 3=*DE/current-to-rand/1*), the control parameters to be set are N, CR, F, K and G_{max}. We choose from $N = 20P$, $N = 10P$ and $N = 2P$ to obtain large, intermediate and small populations. CR is 0.99, while $F = 0.6$ is used unless otherwise stated. Since the purpose of this study is the search for a global optimum for the lot sizes in ARP, $G_{max} = 100{,}000$ is a reasonable stopping criteria based on the results below. If the optimum is not found after 100,000 iterations, we can conclude that DE is underperforming with the control parameters used. For the second stopping criteria, we allow a small deviation $\epsilon = 10^{-7}$ between the best and the worst population member as required by $\left|\frac{\text{Best member - Worst member}}{\text{Worst member}}\right| < \epsilon$. This involves that all members of the population must converge to the same solution before the algorithm is ended. It ensures high quality results.

The research questions as described in Sect. 1 are answered below. To this end, we have collected data from a large manufacturing system with globally dispersed but cooperating members that are active in the area of industrial tools. As a result, these datasets represent SCMS from practice with realistic problem sizes. We distinguish four ARP problems that differ in size with respect to the number of products P, resources R and operations O_p. See Table 1 for their dimension. It will be clear that DE can be applied to lot size decisions in large real-life stochastic SCMS. For each ARP problem in this table and each instance of DE, 15 independent DE runs have been performed. All calculations are executed on a 3.7 GHz Quad-Core Intel Xeon E5 processor, 16 GB 1,866 MHz DDR3 ECC memory. The computational effort is expressed in minutes CPU time. Tables 5, 6, 7, 8, 9 and 10 in the Appendix summarize these results. When the format is italic, the number of generations has reached its maximum G_{max}, an indication that the optimization result is not reliable yet. When the format is bold, it represents a first-best performing DE setting.

Table 1 Dimension of ARP problems

ARP	1	2	3	4
P	100	50	25	10
$\sum_p O_p$	640	424	216	86
R	133	133	114	86

Table 2 Base setting of DE relative to SD

ARP	DE		SD
	Large population	Intermediate population	
Lead time			
Objective			
1	4,494	4,494	4,691 (+4.38 %)
2	4,682	4,682	4,746 (+1.37 %)
3	3,630	3,630	3,698 (+1.87 %)
4	2,440	2,440	2,469 (+1.19 %)
CPU time			
1	9,850	2,634	440
2	846	250	22.4
3	46.1	18.7	12.0
4	1.17	0.53	0.50
Cost			
Objective			
1	95,326	95,326	98,805 (+3.65 %)
2	86,330	86,330	86,434 (+0.12 %)
3	69,424	69,424	69,467 (+0.06 %)
4	50,918	50,918	50,939 (+0.04 %)
CPU time			
1	10,931	2,872	32.6
2	875	260	4.11
3	46.48	18.77	0.67
4	1.25	0.58	0.28

Research Question 1 *Do standard schemes of the differential evolution algorithm exist to optimize large real-life ARP problems efficiently in a SCMS?*

For a base DE setting (Scheme 1, $F = 0.6$) and relative large population sizes, Table 2 shows the results for the lead time and the cost functions [Eqs. (10) and (11)] as well as CPU time in minutes. In each case where the DE setting leads to the optimal solution, no variation in the objective measure is observed between multiple independent DE runs because the underlying lot size values always converge to exactly the same set. Consequently, we are confident that such a solution is near the global optimum. Also, displaying average values of 15 independent DE runs is sufficient here. Overall, CPU time is rapidly increasing with the decision variables P and the population size N, especially when the cost objective is considered.

Table 3 Measures on the absolute deviations between the optimal lot size quantities found by DE and SD

Function	Lead time				Cost			
ARP	1	2	3	4	1	2	3	4
Mean	9.16	0.38	0.36	0.40	9.14	0.48	0.52	0.50
Variance	2,383	0.24	0.23	0.24	655	0.29	0.25	0.25
Mininmum	0	0	0	0	1	0	0	0
Maximum	404	1	1	1	213	2	1	1

In order to give an idea about the DE performance relative to SD as the traditional solution method for ARP, the best solution found by SD and deviations from the DE result in terms of percentage are included. A major observation is that the objective value can always be improved by DE, and usually more when P is high and when lead time is considered. Table 3 shows statistical measures (mean, variance, minimum and maximum) on the absolute deviations between the optimal lot size quantities found by DE and SD, for all products p in a specific ARP problem. The lot size values according to DE clearly deviate more from SD when the number of decision variables is high. The next step is to find more efficient parameter settings for DE.

The more the CPU time can be improved for DE to find the optimal solution, the more appropriate this management decision tool becomes at the intermediate planning horizon. First, we dramatically reduce the population size to $N = 2P$, followed by an extensive study with different values for F and K that are applied to the three schemes as described in Sect. 4. Note that CR and G_{max} remain at their initial value. Tables in the Appendix give an overview of the results, expressed as an increase (+) or decrease (−) in terms of percentage relative to the DE results in Table 2 with an intermediate population (i.e. $N = 10P$). A quality level number is included to distinguish five groups of DE parameter settings with respect to objective value and DE performance.

- Quality level 5 quickly converges to inferior solutions. It typically occurs for low F values, especially in combination with low P values and the objective of lead time minimization. When Scheme 3 is used with $K = 1$, it occurs for any value of F.
- Quality level 4 is not able to find good solutions after 100,000 generations. Due to this very slow convergence rate, these settings are not suitable. It typically occurs for high F values in Scheme 2 and high F values in Scheme 3 with $K = 0$, especially in combination with high P values. When Scheme 3 is used with $K = 0.5$, it usually occurs for any value of F.
- Quality level 3 is able to find good solutions, but the population is still diversified after 100,000 generations. This is an indication of either stagnation or a slow convergence rate. Therefore, these settings are not usable in practice because either no information about the global solution is given to know when to quit the search process or it takes too long before all the population members are converged to the optimal solution. It typically occurs for intermediate F values in Scheme 2 and for any F value in Scheme 3 with $K = 0$, especially in combination with low P values.
- Quality level 2 is able to find solutions that are close to the global optimum. These settings can be justified when CPU time is critical. It typically occurs for intermediate to high F values in Scheme 1, for $F = 0.4$ in Scheme 2 and for intermediate to high F values in Scheme 3 with $K = [0, 1]$.
- Quality level 1 is exactly equal to the solution from Table 2, both from an objective function and a lot size point of view. The only setting where this occurs for all the ARP problems in this study is the DE/current-to-rand/1 scheme with $F = 0.6$ and $K = [0, 1]$.

Table 4 DE results for ARP: CPU time of best DE control parameter set relative to the base setting with an intermediate population

	ARP			
	1 (%)	2 (%)	3 (%)	4 (%)
Lead time	92.26	95.75	−94.38	−84.38
Cost	93.63	95.29	−94.23	−85.71

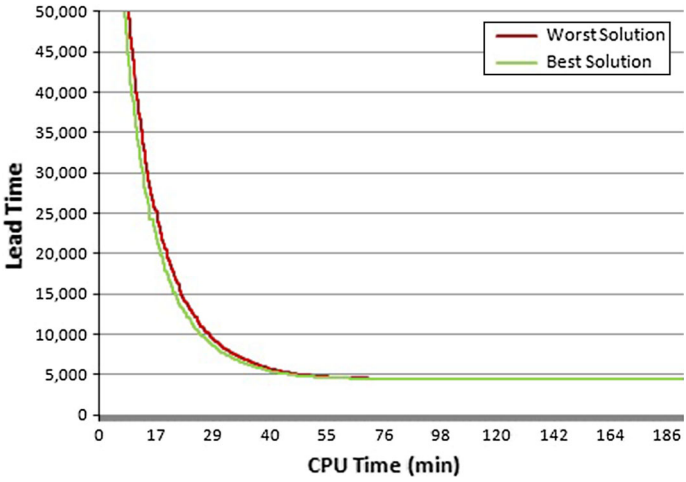

Fig. 4 DE results for ARP: convergence speed of the lead time for the best DE control parameter set

The significant reduction in CPU time makes it the most preferable DE control parameter set for the ARP problems in this study. Its results relative to the base setting (i.e. Table 2 with $N = 10P$) are summarized in Table 4. Since there is no guarantee that this DE control parameter set is absolutely the best for any ARP problem in general, we are confident that based on the outcome of this extensive study, it can serve as a good initial set.

For this DE setting (quality level 1), Fig. 4 gives an idea about the convergence speed, where the improvement of the best and the worst population member with respect to lead time is plotted as a function of CPU time. The gap between the best and the worst solution, as well as further lead time reductions become smaller after 50 min. Figure 5 shows that DE detects a solution that is within 99.96 % of the range between the best initial solution and the global optimum only after 30 % of total CPU time. This means that 70 % of the processor time is dedicated to minor improvements. This is an interesting managerial insight for an implementation in practice: good solutions that are close to the global optimum can already be found within an acceptable fraction of total CPU time.

Conclusion 1 *DE is effective in the search for the global optimal lot size values in ARP problems of realistic size. The improved performance relative to SD is more pronounced when more products are involved. We suggest to use the DE/current-to-rand/1 scheme with* $F = 0.6$ *and* $K \in [0, 1]$ *as an initial efficient control parameter set for ARP. Close to optimal solutions can be found quickly.*

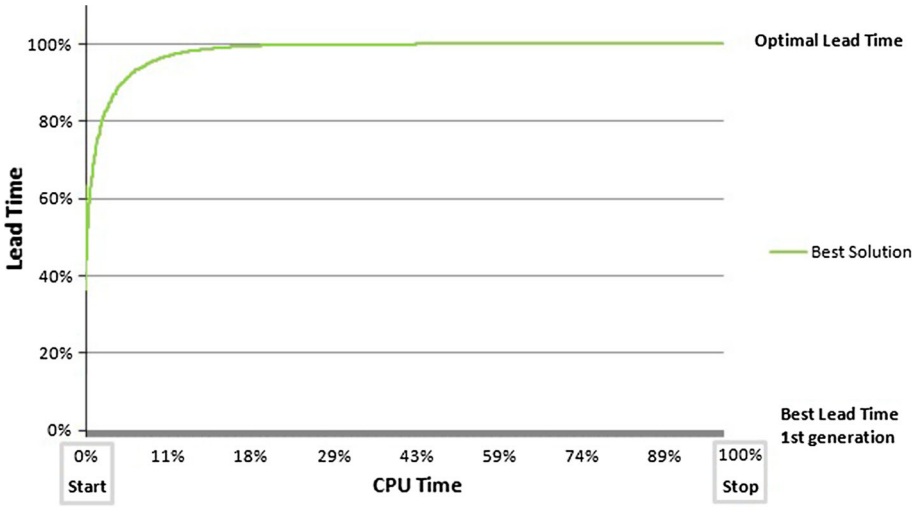

Fig. 5 DE results for ARP: proportional convergence speed of the lead time for the best DE control parameter set

Research Question 2 *Is there evidence for the generally postulated convex relationship between lot size and lead time in a multi-product, multi-resource SCMS?*

From the extensive study above, other important observations can be derived. First of all, we see that the optimal solutions found by DE with $N = 20P$ and $N = 10P$ can also be detected with $N = 2P$. Referring to the guidelines for the DE control parameters in Sect. 4.1.4, we can interpret the required value of N as an indicator of model complexity in terms of the number of local optima that exist. Values up to $100P$ are not uncommon for complex functions with many local optima. Since tables in the Appendix show that a population size $N = 2P$ is often large enough to find the best solution, we can conclude that the lot size model has convexity properties. The consistent lot size values found by multiple independent DE runs in the previous research question is another indication that smooth objective functions exist in the ARP model. A second observation is that both objective functions are rather flat because incremental changes in the lot size values only contribute to an incremental deterioration of the objective value. This corresponds to what we have seen in practice. It is an insight of great value for managerial purposes: some robustness of the search process may be sacrificed in order to promote a faster convergence rate. Another remark is that deviations from the optimal objective values are smaller for the cost function, and that the convergence rate for this objective function is faster. This means that it is easier to find good lot size values for this objective than for lead time reduction, which can be explained by the staircase function. Finally, it is logic that the optimal lot sizes differ for both objective functions because the impact of lot size changes is measured differently (time and costs). Under cost minimization, the lot sizes are usually larger for many products p in our case study, while only a few products p have smaller lot sizes. This can be explained by the cost structure that is used in this study. It penalizes setting up resources relatively more than the lead time model.

Conclusion 2 *Results and characteristics of the DE search process provide evidence that a convex relationship exists between the lot size and the lead time in ARP.*

6 Conclusion

An Advanced Resource Planning (ARP) problem that applies to stochastic supply chain management systems (SCMS) with multiple products and multiple resources is modeled as a queueing network, while integrating parallel servers, multi-period resource schedules and variabilities from rework and breakdown. We have developed two objective functions, one for the overall expected lead time and one for the overall expected costs related to setting up resources and holding work-in-process. Both functions depend on the lot size values for multiple products that are processed in a sequence of operations on one or more resources, possibly performed at multiple, interdependent supply chain members. The analytical model can be used to select appropriate lot sizes in these complex supply chain networks.

For different problems with practical relevance, we have shown that in search of the optimal lot sizes, the differential evolution method (DE) as a member of the genetic algorithms always outperforms the steepest descent algorithm, the method that is commonly used to solve the lot size problem. This is particularly the case when many products are present.

We have also shown that DE is able to detect lot size solutions near the global optimum because different DE control parameter settings in combination with large population sizes always converge to the same set of lot size values in each ARP problem that is investigated in the case study.

The speed of its search process can be enhanced without a reduction in the solution quality by using a smaller population size of twice the number of decision variables in combination with appropriate control parameters. We suggest to use DE/current-to-rand/1 with $F = 0.6$ and K randomly chosen between 0 and 1 as the best performing initial control setting if a relatively fast convergence rate towards the global optimum is desired in ARP. In addition, close to optimal solutions are found within a small fraction of total CPU time.

A population that converges towards the global optimum with a size of only twice the number of decision variables is only possible when the objective function is relatively smooth without many local optima. Since this is the case in ARP, evidence is provided that the convex relationship between the lot size and the objective of lead time or cost minimization holds in a complex SCMS.

These findings are of great value for practitioners as well: large scale SCMS with globally dispersed but cooperating members can use ARP in combination with the DE optimizer to find lot sizes within an acceptable time limit that further improve their lead time and delivery performance.

Another outcome for managers is to be aware that a different view (time vs. cost, or equivalently flow vs. finance) will lead to a different optimal set of lot size values. As a member of genetic algorithms, DE is highly suitable to handle multi-criteria analyses, a useful approach to combine these two views. It creates an opportunity for further research where the lot size impact on lead time and total cost is jointly considered.

Acknowledgments The authors would like to thank the Fund for Scientific Research Flanders (FWO) for their support with the Post Doctoral Research project for Kris Lieckens.

Appendix: Results for Control Parameter Settings in DE

See Tables 5, 6, 7, 8, 9 and 10.

Table 5 Results for DE/rand/1/bin with a small population compared to the base setting with an intermediate population

ARP	*F*				
	0.2	0.4	0.6	0.8	0.99
Lead time					
Objective (quality)					
1	2,453 % (5)	0.89 % (2)	**0 % (1)**	0.04 % (2)	0.30 % (2)
2	2,966 % (5)	18.65 % (5)	0.04 % (2)	**0 % (1)**	0.14 % (2)
3	3,517 % (5)	232 % (5)	0.45 % (2)	**0 % (1)**	**0 % (1)**
4	2,826 % (5)	1,560 % (5)	0.46 % (2)	0.17 % (2)	0.07 % (2)
CPU time					
1	−99.72 %	−98.50 %	−94.89 %	−82.15 %	−78.19 %
2	−99.46 %	−97.16 %	−94.36 %	−79.46 %	−81.47 %
3	−98.15 %	−93.44 %	−92.34 %	−78.21 %	−77.09 %
4	−95.39 %	−79.97 %	−76.13 %	−74.96 %	−68.65 %
Cost					
Objective (quality)					
1	653 % (5)	0.71 % (2)	**0 % (1)**	0.02 % (2)	0.30 % (2)
2	543 % (5)	2.25 % (5)	**0 % (1)**	**0 % (1)**	0.14 % (2)
3	252 % (5)	52.73 % (5)	0.01 % (2)	**0 % (1)**	**0 % (1)**
4	137 % (5)	79.88 % (5)	0.14 % (2)	0.07 % (2)	0.29 % (2)
CPU time					
1	−99.41 %	−98.83 %	−94.62 %	−79.52 %	−78.55 %
2	−99.52 %	−97.16 %	−93.87 %	−77.72 %	−74.26 %
3	−98.37 %	−87.56 %	−90.66 %	−76.85 %	−66.27 %
4	−97.13 %	−87.72 %	−82.25 %	−74.48 %	−62.46 %

Table 6 Results for DE/rand/2/bin with a small population compared to the base setting with an intermediate population

ARP	*F*				
	0.2	0.4	0.6	0.8	0.99
Lead time					
Objective (quality)					
1	416 % (5)	**0 % (1)**	0.04 % (2)	2,021 % (4)	2,238 % (4)
2	2,156 % (5)	0.04 % (2)	0.05 % (3)	725 % (4)	1,348 % (4)
3	2,082 % (5)	1.38 % (5)	**0 % (1)**	0.16 % (3)	177 % (4)
4	2,254 % (5)	26.75 % (5)	**0 % (1)**	**0 % (1)**	2.38 % (5)
CPU time					
1	−99.03 %	−94.96 %	−95.31 %	*−60.07 %*	*−64.27 %*
2	−98.40 %	−94.68 %	*29.84 %*	*99.89 %*	*59.64 %*
3	−96.87 %	−91.82 %	−42.54 %	*339 %*	*440 %*
4	−92.89 %	−77.98 %	−72.88 %	−38.10 %	−7.43 %

Table 6 continued

ARP	*F*				
	0.2	0.4	0.6	0.8	0.99
Cost					
Objective (quality)					
1	230 % (5)	0.01 % (2)	**0 % (1)**	826 % (4)	926 % (4)
2	388 % (5)	**0 % (1)**	0 % (3)	207 % (4)	298 % (4)
3	265 % (5)	0.03 % (2)	**0 % (1)**	0 % (3)	11.19 % (4)
4	62.94 % (5)	0.04 % (2)	**0 % (1)**	**0 % (1)**	0.13 % (2)
CPU time					
1	−98.76 %	−95.59 %	−95.41 %	*−59.86 %*	*−60.36 %*
2	−98.17 %	−94.64 %	*89.22 %*	*98.81 %*	*95.40 %*
3	−97.95 %	−91.05 %	−36.90 %	*725 %*	*555 %*
4	−93.93 %	−74.68 %	−66.04 %	−3.15 %	11.31 %

Table 7 Results for DE/current-to-rand/1 with a small population and $K = 0$ compared to the base setting with an intermediate population

ARP	*F*				
	0.2	0.4	0.6	0.8	0.99
Lead time					
Objective (quality)					
1	3.28 % (4)	69.63 % (4)	388 % (4)	1,022 % (4)	1,532 % (4)
2	0.11 % (3)	1.64 % (4)	13.02 % (4)	138 % (4)	303 % (4)
3	0 % (3)	0 % (3)	0 % (3)	0.29 % (3)	3.35 % (4)
4	3,744 % (5)	4,036 % (5)	2,580 % (5)	366 % (5)	0.15 % (2)
CPU time					
1	*−70.00 %*	*−90.78 %*	*−83.26 %*	*−81.98 %*	*−78.79 %*
2	*82.39 %*	*42.54 %*	*−40.36 %*	*−31.30 %*	*−12.32 %*
3	*725 %*	*664 %*	*722 %*	*472 %*	*152 %*
4	33.51 %	33.77 %	35.65 %	36.42 %	−42.64 %
Cost					
Objective (quality)					
1	0.75 % (3)	20.47 % (4)	281 % (4)	567 % (4)	695 % (4)
2	0.03 % (3)	0.26 % (3)	2.62 % (4)	14.75 % (4)	56.89 % (4)
3	0 % (3)	0 % (3)	0 % (3)	0.03 % (3)	0.29 % (3)
4	242 % (5)	146 % (5)	125 % (5)	5.53 % (5)	0.02 % (2)
CPU time					
1	*−57.48 %*	*−85.31 %*	*−73.84 %*	*−73.67 %*	*−73.96 %*
2	*115.57 %*	*93.93 %*	*−2.85 %*	*2.65 %*	*15.49 %*
3	*852 %*	*859 %*	*900 %*	*847 %*	*520 %*
4	76.88 %	77.88 %	93.42 %	90.68 %	−65.60 %

Table 8 Results for DE/current-to-rand/1 with a small population and $K = [0, 1]$ compared to the base setting with an intermediate population

ARP	*F*				
	0.2	0.4	0.6	0.8	0.99
Lead time					
Objective (quality)					
1	4,362 % (5)	339 % (5)	**0 % (1)**	110 % (4)	900 % (4)
2	4,035 % (5)	578 % (5)	**0 % (1)**	**0 % (1)**	19.32 % (4)
3	3,838 % (5)	635 % (5)	**0 % (1)**	**0 % (1)**	**0 % (1)**
4	3,176 % (5)	928 % (5)	**0 % (1)**	**0 % (1)**	**0 % (1)**
CPU time					
1	−99.83 %	−99.20 %	−92.26 %	*−80.02 %*	*−67.87 %*
2	−99.45 %	−98.08 %	−95.75 %	−1.80 %	*−25.05 %*
3	−98.16 %	−96.00 %	−94.38 %	−60.29 %	225 %
4	−90.92 %	−86.16 %	−84.38 %	−74.85 %	−59.93 %
Cost					
Objective (quality)					
1	1,316 % (5)	202 % (5)	**0 % (1)**	32.10 % (4)	400 % (4)
2	664 % (5)	130 % (5)	**0 % (1)**	**0 % (1)**	1.11 % (4)
3	279 % (5)	88.67 % (5)	**0 % (1)**	**0 % (1)**	**0 % (1)**
4	147 % (5)	38.20 % (5)	**0 % (1)**	**0 % (1)**	**0 % (1)**
CPU time					
1	−99.87 %	−99.20 %	−93.63 %	*−74.01 %*	*−63.83 %*
2	−99.39 %	−98.33 %	−95.29 %	80.04 %	*4.88 %*
3	−98.33 %	−95.92 %	−94.23 %	−66.54 %	117 %
4	−92.54 %	−87.67 %	−85.71 %	−77.65 %	−46.48 %

Table 9 Results for DE/current-to-rand/1 with a small population and $K = 0.5$ compared to the base setting with an intermediate population

ARP	*F*				
	0.2	0.4	0.6	0.8	0.99
Lead time					
Objective (quality)					
1	4,925 % (4)	4,746 % (4)	652 % (5)	5.52 % (4)	503 % (4)
2	5,137 % (4)	4,506 % (4)	1,544 % (4)	2.49 % (4)	1.05 % (4)
3	5,053 % (4)	4,858 % (4)	2,445 % (4)	11.35 % (4)	0.13 % (3)
4	3,448 % (5)	4,171 % (4)	1,735 % (4)	15.44 % (4)	3.48 % (4)
CPU time					
1	*−39.37 %*	*−39.37 %*	−96.36 %	*−56.29 %*	*−70.69 %*
2	*131.94 %*	*131.13 %*	*131.92 %*	*127.56 %*	*58.22 %*
3	*919 %*	*925 %*	*913 %*	*878 %*	*885 %*
4	−61.76 %	*7,736 %*	*7,687 %*	*7,290 %*	*7,629 %*

Table 9 continued

ARP	*F*				
	0.2	0.4	0.6	0.8	0.99
Cost					
Objective (quality)					
1	1,488 % (4)	1,301 % (4)	544 % (4)	2.18 % (4)	224 % (4)
2	775 % (4)	764 % (4)	397 % (4)	7.37 % (4)	1.06 % (4)
3	439 % (4)	413 % (4)	275 % (4)	7.55 % (4)	0.01 % (2)
4	232 % (4)	165 % (4)	94.18 % (4)	13.64 % (4)	**0 % (1)**
CPU time					
1	*−45.75 %*	*−45.75 %*	*−45.75 %*	*−53.56 %*	*−63.57 %*
2	*142.12 %*	*142.12 %*	*142.12 %*	*140.03 %*	*105.64 %*
3	*932 %*	*931 %*	*933 %*	*930 %*	33.33 %
4	*6,815 %*	*6,815 %*	*6,814 %*	*6,813 %*	−52.87 %

Table 10 Results for DE/current-to-rand/1 with a small population and $K = 1$ compared to the base setting with an intermediate population

ARP	*F*				
	0.2	0.4	0.6	0.8	0.99
Lead time					
Objective (quality)					
1	4,030 % (5)	2,030 % (5)	212 % (5)	48.88 % (5)	318 % (5)
2	4,233 % (5)	1,640 % (5)	325 % (5)	28.88 % (5)	196 % (5)
3	3,799 % (5)	2,011 % (5)	359 % (5)	20.64 % (5)	43.89 % (5)
4	3,114 % (5)	2,943 % (5)	1,022 % (5)	308 % (5)	159 % (5)
CPU time					
1	−99.81 %	−97.82 %	−88.72 %	−63.95 %	−85.90 %
2	−99.51 %	−97.56 %	−90.49 %	−66.73 %	−79.30 %
3	−99.03 %	−94.90 %	−78.94 %	−68.17 %	−67.12 %
4	−94.77 %	−87.32 %	−67.30 %	−50.17 %	−70.40 %
Cost					
Objective (quality)					
1	1,276 % (5)	763 % (5)	140 % (5)	38.07 % (5)	199 % (5)
2	691 % (5)	396 % (5)	166 % (5)	5.75 % (5)	28.35 % (5)
3	350 % (5)	216 % (5)	95.84 % (5)	2.79 % (5)	11.94 % (5)
4	156 % (5)	138 % (5)	66.98 % (5)	7.51 % (5)	0.78 % (2)
CPU time					
1	−99.86 %	−97.98 %	−93.56 %	−75.07 %	−87.80 %
2	−99.47 %	−96.96 %	−92.64 %	−69.80 %	−82.27 %
3	−98.71 %	−94.44 %	−84.04 %	−72.34 %	−74.32 %
4	−92.53 %	−88.54 %	−68.04 %	−68.73 %	−78.22 %

References

Albin, S. (1981). *Approximating queues with superposition arrival processes*. PhD thesis, Columbia University, Department for Industrial Engineering and Operations Research.

Babu, B., & Angira, R. (2002). A differential evolution approach for global optimization of MINLP problems. In *Proceedings of 4th Asia-Pacific conference on simulated evolution and learning* (vol. 2, pp. 880–884).

Bitran, G., & Tirupati, D. (1988). Multiproduct queueing networks with deterministic routing: Decomposition approach and the notion of interference. *Management Science*, *34*(1), 75–100.

Buzacott, J., & Shanthikumar, J. (1985). On approximate queueing models of dynamic job shops. *Management Science*, *31*, 870–887.

Buzacott, J., & Shanthikumar, J. (1993). *Stochastic models of manufacturing systems*. Englewood Cliffs, NY: Prentice Hall.

Hopp, W., & Spearman, M. (2000). *Factory physics*. New York: McGraw-Hill.

Karmarkar, U. (1987). Lot sizes, lead times and in-process inventories. *Management Science*, *33*, 409–423.

Karmarkar, U., Kekre, S., & Kekre, S. (1985a). Lot sizing in multi-item multi-machine job shops. *IIE Transactions*, *17*(3), 290–292.

Karmarkar, U., Kekre, S., Kekre, S., & Freeman, S. (1985b). Lot-sizing and lead-time perfromance in a manufacturing cell. *Interfaces*, *15*(2), 1–9.

Karmarkar, U., Kekre, S., & Kekre, S. (1992). Multi-item batching heuristics for minimization of queueing delays. *European Journal of Operational Research*, *58*, 99–111.

Kuik, R., & Tielemans, P. (2004). Expected time in system analysis of a single-machine multi-item processing center. *European Journal of Operational Research*, *156*, 287–304.

Lambrecht, M., Ivens, P., & Vandaele, N. (1998). ACLIPS: A capacity and lead time integrated procedure for scheduling. *Management Science*, *44*(11), 1548–1561.

Lampinen, J. (2001). Multi-constrained nonlinear optimization by the differential evolution algorithm. In *6th On-line world conference on soft computing in industrial applications (WSC6)*.

Lampinen, J., & Zelinka, I. (1999). Mechanical engineering design optimization by differential evolution (chap 8, pp. 127–146). London: McGraw-Hill.

Little, J. D. C. (1961). A proof of the queuing formula: L = aw. *Operations Research*, *9*(3), 383–387.

Nieuwenhuyse, I. V., Boeck, L. D., Lambrecht, M., & Vandaele, N. (2011). Advanced resource planning as a decision support module for erp. *Computers in Industry*, *62*(1), 1–8.

Nieuwenhuyse, I. V., & Vandaele, N. (2006). The impact of delivery lot splitting on delivery reliability in a two-stage supply chain. *International Journal of Production Economics*, *104*(2), 694–708.

Nieuwenhuyse, I. V., Vandaele, N., Rajaram, K., & Karmarkar, U. (2007). Buffer sizing in multi-product multi-reactor batch processes: Impact of allocation and campaign sizing policies. *European Journal of Operational Research*, *179*(2), 424–443.

Pahl, J., Voss, S., & Woodruff, D. L. (2007). Production planning with load dependent lead times: An update of research. *Annals of Operations Research*, *153*(1), 297–345.

Shanthikumar, J., & Buzacott, J. (1981). Open queueing network models of dynamic job shops. *International Journal of Production Research*, *19*(3), 255–266.

Shanthikumar, J., & Sumita, U. (1988). Approximations for the time spent in a dynamic job shop with applications to due-date assignment. *International Journal of Production Research*, *26*, 1329–1352.

Storn, R., & Price, K. (1997). Differential evolution—A simple end efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, *11*, 341–359.

Vandaele, N. (1996). *The impact of lot sizing on queueing delays: Multi product, multi machine models*. PhD thesis, Katholieke Universiteit Leuven.

Vandaele, N., Boeck, L. D., & Callewier, D. (2002). An open queueing network for lead time analysis. *IIE Transactions*, *34*, 1–9.

Vandaele, N., Vannieuwenhuyse, I., & Cupers, S. (2003). Optimal grouping for a nuclear magnetic resonance scanner by means of an open queueing model. *European Journal of Operational Research*, *151*(1), 181–192.

Whitt, W. (1993). Approximations for the GI/G/m queue. *Production and Operations Management*, *2*(2), 114–161.

Zhang, J., & Sanderson, A. (2009). Jade: Adaptive differential evolution with optional external archive. *IIE Transactions on Evolutionary Computation*, *13*(5), 945–958.

Zhou, Z., & Guan, Y. (2013). Two-stage stochastic lot-sizing problem under cost uncertainty. *Annals of Operations Research*, *209*(1), 207–30.